IN THE CLAIMS:

Please amend the claims as follows:

Claim 1 (Currently Amended) A method for execution by a data processor for indexing and storing vertex data associated with the vertices that define neighboring primitives to enhance primitive processing by the processor, comprising:

selecting a reference vertex;

identifying one-ring neighbor vertices of the reference vertex;

assigning a unique reference to each of the one-ring neighbor vertices;

assigning a unique neighbor index to each of the one of the one-ring neighbor vertices in a sequential order around the reference vertex;

storing the neighboring primitives associated with the one-ring neighbor vertices based on the assigned neighbor indexes, wherein the unique neighbor index includes a [[an]] user-specified offset relative to each of the neighboring primitives, the offset being used to specify a consistent order of calculation of the neighboring primitives by ordering the one-ring neighbor vertices for use during primitive processing; and

displaying the primitives.

Claims 2 – 4 (Cancelled)

Claim 5 (Previously Presented): A method for indexing as in Claim 33, wherein the primitive is a quadrilateral primitive.

Claim 6 (Previously Presented): A method for indexing as in Claim 31, wherein the polygonal primitive is a triangular primitive.

Claim 7-8 (Cancelled):

Claim 9 (Original): A method for indexing as in Claim 1, wherein the at least one primitive defines a volume.

Claims 10 – 11 (Cancelled)

Claim 12 (Original): A method for indexing as in Claim 1, further comprising identifying an edge between a first vertex and a second vertex, the second vertex being a one-ring neighbor of the first vertex.

Claim 13 (Original): A method for indexing as in Claim 12, further comprising:
assigning the unique reference of the first vertex to the edge; and
assigning the unique neighbor index of the second vertex to the edge.

Claim 14 (Original): A method for indexing as in Claim 13, further comprising:
assigning the unique reference of the second vertex to the edge; and
assigning the unique neighbor index of the first vertex to the edge.

Claim 15 (Currently Amended): A ~~computer-readable medium having stored thereon a~~ software program, for execution on a graphics processing unit (GPU), that includes instructions that when executed by the GPU cause the GPU to index ~~determining a method for execution by a data processor for indexing~~ vertex data for data input to a graphics program to enhance vertex processing~~, comprising~~ by performing the steps of:
identifying a vertex;
assigning a reference to the vertex to define a first reference vertex;
identifying one-ring neighbor vertices of the vertex;
assigning the reference to each of the one-ring neighbor vertices identified;
assigning a neighbor index to one of the one-ring neighbor vertices identified;
successively incrementing the neighbor index to provide incremented neighbor indices for assignment to the one-ring neighbor vertices remaining; and
sequentially assigning one of the incremented neighbor indices to each of the one-ring neighbor vertices remaining, wherein the ordering of the one-ring neighbor vertices is user determined.

Claim 16-17 (Cancelled):

Claim 18 (Previously Presented): A computer readable medium as in Claim 15, further comprising:

totaling the one-ring neighbor vertices sharing an edge with the vertex to provide a total; and

indicating the total as a valence of the vertex.

Claim 19 (Previously Presented): A computer readable medium as in Claim 15, wherein the neighbor index includes an offset.

Claim 20 (Previously Presented): A computer readable medium as in Claim 15, wherein the vertex and at least a portion of the one-ring neighbors define a primitive.

Claim 21 (Previously Presented): A computer readable medium as in Claim 15, further comprising:

assigning a second reference to one of the one-ring neighbor vertices to define a second reference vertex;

identifying one-ring neighbor vertices of the second reference vertex, the one-ring neighbor vertices including the first reference vertex;

assigning the second reference to each of the one-ring neighbor vertices of the second reference vertex identified;

assigning an additional neighbor index to one of the one-ring neighbor vertices of the second reference vertex identified;

successively incrementing the additional neighbor index to provide incremented additional neighbor indices for assignment to the one-ring neighbor vertices of the second reference vertex; and

sequentially assigning one of the incremented additional neighbor indices to each of the one-ring neighbor vertices of the second reference vertex remaining.

Claim 22 (Previously Presented): A computer readable medium for indexing for indexing as in Claim 21, wherein the first reference vertex may be referenced using the first reference or using the second reference and an additional neighbor index assigned.

Claim 23 (Previously Presented): A computer readable medium for indexing as in Claim 22, wherein data corresponding to the first reference vertex is stored in a portion of memory accessed using the first reference.

Claim 24 (Previously Presented): A computer readable medium for indexing as in Claim 23, wherein data corresponding to the first reference vertex stored in the portion of memory is accessed using the second reference and the additional neighbor index assigned.

Claim 25 (Previously Presented): A computer readable medium for indexing as in Claim 21, wherein the second reference vertex may be referenced using the first reference and a neighbor index assigned or using the second reference.

Claim 26 (Currently Amended): A computer readable medium as in claim 1 wherein edges are referenced in relation to a selected vertex of an originating primitive and one neighbor vertex of the selected reference vertex.

Claim 27 (Previously Presented): A method for execution by a data processor for indexing vertex data defining at least one primitive to enhance primitive processing by the processor, comprising:

assigning a unique reference to each vertex defining the at least one primitive;

identifying one-ring neighbor vertices of each vertex;

assigning the unique reference to each of the one-ring neighbor vertices of each vertex;

assigning a unique neighbor index to each of the one of the one-ring neighbor vertices of each vertex wherein a unique neighbor index includes a user-specified offset to specify an order of calculation in primitive processing: and

storing the primitives for display.

Claim 28 (Previously Presented): The method of claim 27 wherein the offset is determined by the number of one-ring neighbor vertices to a reference vertex.

Claim 29 (Previously Presented): The method of claim 28 wherein the offset determines the sequence with which the neighboring primitives are processed.

Claim 30 (Previously Presented): A method as in claim 1 wherein one or more vertex of the primitive is stored in only one location and is accessible in more than one way

based on one of the unique neighbor indexes.

Claim 31 (Previously Presented): A method as in claim 30 including storing a valence for the reference vertex defining the number of one-ring neighbor vertices for the reference vertex.

Claim 32 (Previously Presented): A method as in claim 31 wherein direction of ordering of neighbor indexes is user specified.

Claim 33 (Previously Presented): A method as in clam 31 wherein the valence is the number of vertices that share an edge with the reference vertex and is stored with a modulo of the reference vertex to determine the neighbor index, where modulo is a total number of one-ring neighbors of one of the neighbor vertices.

Claim 34 (Previously Presented): A method as in claim 6 wherein edge data defining each edge of the triangular primitive is referenced in one of two ways.

Claim 35 (Previously Presented): A method as claimed in claim 5 wherein the indexing of neighboring quadrilateral primitives is redundant but that storage of the index data is not redundant.

Claim 36 (Previously Presented): A method as claimed in claim 1 including storing edge data for each neighbor primitive to a reference primitive, wherein only edges adjacent to or shared by a user selected reference vertex of the reference primitive are defined as one of the neighbor primitives.

Claim 37 (Previously Presented): A method as claimed in claim 36 wherein the offset is used to reference control points related to the stored vertex data, wherein the control point includes at least one attribute chosen from the groups consisting of position coordinates and texture coordinates.